PATENT ABSTRACTS OF JAPAN

(11)Publication number : 11-085826
(43)Date of publication of application : 30.03.1999

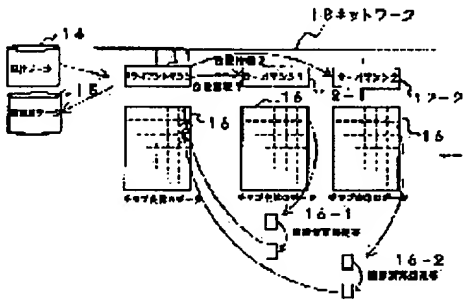
(51)Int.Cl. G06F 17/50
G06F 15/16
G06T 1/20
G06T 11/00
H01L 21/027

(21)Application number : 09-246580
(22)Date of filing : 11.09.1997

(71)Applicant : SONY CORP
(72)Inventor : TAKENOUCHI TAKASHI
SATO YUTAKA
ASHIDA ISAO

(54) PLOTTING DATA PROCESSING SYSTEM AND ITS METHOD

(57)Abstract:
PROBLEM TO BE SOLVED: To improve the efficiency of processing by reducing the transfer load of a network between a client and a server.
SOLUTION: At a data processing system for a plotting device, provided with a parallel processing function by plural work stations connected through a network 13 or the plural servers 12 of a client/serve model by a personal computer, data of the entire chips is transferred in advance from a client to the servers 12, a client 11 transfers designated information of a part desired to process to the servers 12, the servers 12 extract graphic data of a pertinent part from the entire data already transferred to execute graphic arithmetical processing, etc., and transfers the result to the client 11, and the client 11 merges the result to prepare a final data file for plotting 15.



LEGAL STATUS

[Date of request for examination]

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number]

[Date of registration]

[Number of appeal against examiner's decision of rejection]

(19)日本国特許庁（ＪＰ）　(12) **公開特許公報（A）**　(11)特許出願公開番号

特開平11－85826

(43)公開日　平成11年(1999)３月30日

(51)Int.Cl.[6]		識別記号	FI		
Ｇ０６Ｆ	17/50		Ｇ０６Ｆ	15/60	６０４Ｚ
	15/16	３７０		15/16	３７０Ｎ
Ｇ０６Ｔ	1/20			15/60	６５８Ｐ
	11/00			15/66	Ｋ
Ｈ０１Ｌ	21/027			15/72	３５０

審査請求　未請求　請求項の数６　ＯＬ　（全 15 頁）　最終頁に続く

(21)出願番号　特願平9－246580

(22)出願日　平成９年(1997)９月11日

(71)出願人　000002185
ソニー株式会社
東京都品川区北品川６丁目７番35号

(72)発明者　竹之内　隆司
東京都品川区北品川６丁目７番35号　ソニー株式会社内

(72)発明者　佐藤　裕
東京都品川区北品川６丁目７番35号　ソニー株式会社内

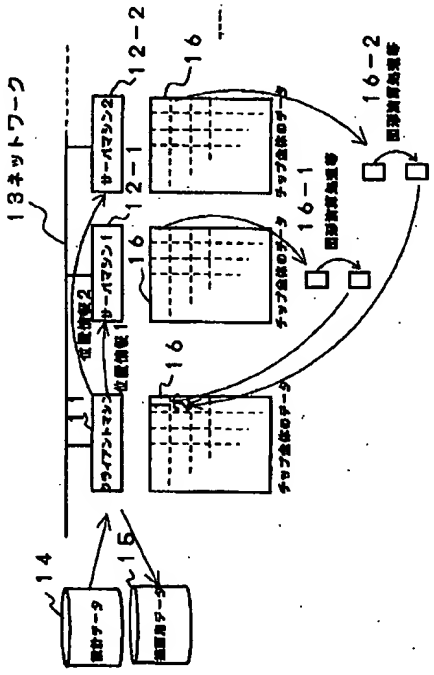
(72)発明者　芦田　勲
東京都品川区北品川６丁目７番35号　ソニー株式会社内

(54)【発明の名称】　描画データ処理システムおよび描画データの処理方法

(57)【要約】

【課題】　クライアントとサーバー間のネットワークの転送負荷を低減して処理の効率化を図る。

【解決手段】　ネットワーク１３で接続された複数のワークステーションまたはパソコンによるクライアント／サーバーモデルの複数のサーバー１２による並列処理機能を備えた描画装置用データ処理システムにおいて、予めチップ全体のデータをクライアント１１からサーバー１２に転送しておき、クライアント１１がサーバー１２に処理させたい部分の指定情報を転送し、サーバー１２は該当部分の図形データを転送済全体データから抽出して図形演算処理等を行い、その結果をクライアント１１に転送し、クライアント１１はその結果をマージして最終的な描画用データファイル１５を作成する。



1

【特許請求の範囲】

【請求項１】　ネットワークで接続された複数のワークステーションまたはパソコンを含んで構成されたクライアント／サーバー・モデルの複数のサーバーによる並列処理機能を備え、パターンデータを処理して描画データを作成する描画データ処理システムにおいて、
予め全体のパターンデータを前記複数のサーバーに転送しておき、クライアントが前記サーバーに処理させたい部分の指定情報を転送し、前記サーバーは前記指定情報で指定された部分の図形データを転送済の前記全体全体のパターンデータから抽出して図形演算処理等を行い、その結果を前記クライアントに転送し、前記クライアントはその結果をマージして最終的な描画用データファイルを作成することを特徴とする描画データ処理システム。

【請求項２】　ネットワークで接続された複数のワークステーションまたはパソコンを含んで構成されたクライアント／サーバー・モデルの複数のサーバーによる並列処理機能を備え、パターンデータを処理して描画データを作成する描画データ処理システムにおいて、
ソフトウェアによるデータの読み込み、データの出力の個々の転送過程で、基本情報単位に対する選択条件と変更条件を指定することにより、前記選択条件と一致する基本情報単位を前記変更条件の内容に応じて変更する選択変更手段を具備することを特徴とする描画データ処理システム。

【請求項３】　パターンデータを処理して描画データを作成する描画データ処理システムにおいて、
計算機上のファイルとして表現されている、繰り返し情報を含むパターンデータ群を、注目したい領域で切り取り抽出する際、抽出領域内に極力もとの前記繰り返し情報を保存するように処理し、前記繰り返し情報を繰り返し単位で処理することを特徴とする描画データ処理システム。

【請求項４】　ネットワークで接続された複数のワークステーションまたはパソコンを含んで構成されたクライアント／サーバー・モデルの複数のサーバーによる並列処理機能を備え、パターンデータを処理して描画データを作成する描画データの処理方法において、
予め全体のパターンデータを前記複数のサーバーに転送しておき、クライアントが前記サーバーに処理させたい部分の指定情報を転送し、前記サーバーは前記指定情報で指定された部分の図形データを転送済の前記全体全体のパターンデータから抽出して図形演算処理等を行い、その結果を前記クライアントに転送し、前記クライアントはその結果をマージして最終的な描画用データファイルを作成することを特徴とする描画データの処理方法。

【請求項５】　ネットワークで接続された複数のワークステーションまたはパソコンを含んで構成されたクライアント／サーバー・モデルの複数のサーバーによる並列

2

処理機能を備え、パターンデータを処理して描画データを作成する描画データの処理方法において、
ソフトウェアによるデータの読み込み、データの出力の個々の転送過程で、基本情報単位に対する選択条件と変更条件を指定することにより、前記選択条件と一致する基本情報単位を前記変更条件の内容に応じて変更することを特徴とする描画データの処理方法。

【請求項６】　パターンデータを処理して描画データを作成する描画データの処理方法において、
計算機上のファイルとして表現されている、繰り返し情報を含むパターンデータ群を、注目したい領域で切り取り抽出する際、抽出領域内に極力もとの前記繰り返し情報を保存するように処理し、前記繰り返し情報を繰り返し単位で処理することを特徴とする描画データの処理方法。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】本発明は、描画データ処理システムおよび描画データの処理方法に関し、特にＬＳＩ設計パターンデータから、ＬＳＩパターンの電子線描画装置用データを作成する描画データ処理システムおよび描画データの処理方法に関する。

【０００２】

【従来の技術】近年、ＬＳＩパターンは微細かつ複雑になって来ており、ＬＳＩ設計パターンデータから、電子線描画装置用データを作成するための変換処理時間が膨大になってきている。この処理時間短縮の一方法として、ネットワークで接続された複数のワークステーションマシンやパソコンを用いたクライアント・サーバーモデルの並列処理システムが用いられている。

【０００３】この並列処理システムは、図１５のような構成となっており、処理フローチャートは図１６のようになっている。図１５で、１１はクライアントマシン、１２－１、１２－２……はサーバーマシン、１３はネットワーク、１４は設計データ、１５は描画用データ、１６はチップ全体のデータ、１６－１、１６－２はチップ全体のデータ１６から分割された各々の部分を示している。

【０００４】このシステムの動作を図１６の処理フローチャートで説明する。まず、設計データ１４からＬＳＩ設計パターンデータを入力し、階層を展開し（ステップ３０１）、そのデータ全体を小さな領域に分割する（ステップ３０２）。個々の分割データを他のマシン１２に転送し（ステップ３０３）、そのマシン１２内で図形演算処理等の実行を行い、その結果を元のマシンに転送する（ステップ３０４）。各マシンから転送された結果をマージし（ステップ３０５）、すべての分割領域の処理が終了したら（ステップ３０６）、最終的な描画装置用データ１５を出力する。以下、分割データを受け取り、演算処理を行うマシンをサーバー１２、データ全体を分

割し、サーバー１２に転送するマシンをクライアント１１と呼ぶこととする。

【０００５】通常この分割単位は、描画装置における描画単位とされることが多いが、全体のチップサイズに比べて微小なため、分割後の領域数が膨大になる。この膨大な領域数に応じて、クライアントとサーバー間にて通信のやり取りを行う回数も膨大となるため、ネットワーク転送負荷が非常に大きくなり、並列処理効率が落ちるという問題がある。

【０００６】また、クライアントは、サーバーにデータを転送する処理と、サーバーからデータを受け取り、それを出力データに追加する処理を行うが、この処理に時間かかかることにより、クライアントが絶えず何らかの処理をすることになれば、サーバーは処理結果をクライアントにデータを転送しても次の処理データをすぐに受け取れないため、待ち時間か生じてしまう。したがって、並列処理効率はクライアントの負荷効率により律速してしまう問題も生じている。このような問題を解決することが望まれている。

【０００７】また、従来、ＬＳＩパターンデータのデータ処理を行いたい場合は、必要に応じたソフトウェアを作成し、それを実行することで対応したり、市販のＣＡＤツールや検証ツールを用いることで対応していた。しかし、前者の場合、前回と違うデータ処理の必要があり、以前作成したソフトウェアで対応できない場合には、ソフトウェアに機能の追加を行ったり、または新たにソフトウェアを作成することで対応しなければならず、そういったことを繰り返すうちに保守管理は困難と増加の度を増し、負担が大きくなる問題をおこしていた。一方、後者の場合、データ処理専用のツールではない為、実行時間が遅かったり、対応できないデータ処理が有るなどの問題があった。

【０００８】ことに、最近のＬＳＩの世代進化に伴い半導体回路の高密度化が要求されており、必然的にＬＳＩパターンデータサイズも増加してきている。そのようなデータ処理を行う為に、処理の種類分のソフトウェアを実行したり、その度にＣＡＤツールや検証ツールに読み込ませることは手間と時間がかかり、ＴＡＴ（Turn Around Time：開発に要する時間）に影響を及ぼす場合も生じるようになってきた。この問題の解決も電子線描画装置用データ作成に取って重要な問題である。

【０００９】さらに、クリップ（抽出）処理に関する問題がある。半導体製造に使用されるマスクを描画するための、ＬＳＩ設計データからマスクの描画データを作成する処理にて、描画データの削減などの目的で、先にも述べたようにデータ全体を小さな領域に分割する。すなわち、ＬＳＩ設計データを特定領域でクリップし、複数の描画データファイルに分ける方法が行なわれている。

【００１０】ここで、クリップ処理の必要性について考える。例えば、マスク上に同一のＬＳＩチップのパターンが複数搭載される場合、全体をそのまま処理することは、データ処理の時間や、作成される描画データサイズの観点から無駄が多い。この様な場合、ＬＳＩチップ部分以外のパターンデータと、ＬＳＩチップ１つ分のパターンデータに分離し、それぞれの描画用パターンファイルを作成して、マスク描画時にＬＳＩチップのパターンを繰り返し描画することで、マスク全体のパターンを描画する。こうすることで、４チップの同一ＬＳＩが搭載されたマスクの場合、全体を一括処理する方法に比べて、実質的に描画データ作成の処理時間と、描画データを約１／４にすることが出来る。

【００１１】同様の例として、メモリを考えてみる。メモリの場合、そのチップの中の多くの面積が、同一のパターンで埋め尽くされている。そこで、繰り返し描画によるマスク描画オーバヘッドが問題にならない範囲で、そのパターンの繰り返し単位を設定し、上記同一ＬＳＩチップと同じ方法で、描画データを作成し、メモリ内同一パターン部分を、設定した繰り返し単位による、繰り返し描画で描画する方法が取られる。このように、描画データ作成時間、描画データサイズの節約のため、クリップ処理が使われている。

【００１２】これ以外の目的でも、例えば、大規模なＬＳＩの特定部分に注目したデータが必要になった場合、全体を処理することは効率が悪いので、必要部分のみを抽出（クリップ）して、データ作成をすることが行なわれている。

【００１３】ところで、このクリップ処理の手順は、入力データの一つ一つに関して、クリップ領域の内部か外部かを判定する。指定領域の外部を残すクリップ処理の場合、領域内部のデータを捨て、外部のデータを残す。領域矩形に交差する図形に関しては、その図形を切り取る等の、図形に対するクリップ処理を行なう。

【００１４】入力データが、繰り返し表現である場合、繰り返し全体で上記繰り返し領域の内部か外部かを判断し、同様に処理するが、繰り返し全体が、領域矩形に交差している場合は、繰り返しを展開したうえで、展開した結果の全てについて、同様の処理を行なうことになる。図１７に繰り返し情報を持つ図形のクリップの例を示す。

【００１５】この例では、繰り返しの図形を全て展開し、４８個の図形全てについて、クリップ領域に対する内／外／交差判定が行なわれる。ここで、先の例に出した、メモリ内部の繰り返しパターン部分のクリップ処理について考えてみると、メモリの最外周部分は、一般に外部のパターンとの連絡の都合上、内部のパターンとは異なる形状をしている場合が多いため、最外周部分を残して、その内部を取り除く必要がある。さらには、最外周を除いた内部のパターンが、繰り返し単位の整数倍の大きさとならない場合は、その余剰となる部分を最外周部分と同様にクリップせずに、残しておかなければなら

ない。

【００１６】規模の大きいメモリーであると、このクリップせずに残すパターン部分の繰り返しが展開されることから、クリップに要する処理時間が長くなり、さらには、クリップ後の出力データサイズが巨大化し、その後の処理の描画データ作成までの処理時間に影響を与えることになる。このため、本来の処理時間の短縮への効果が期待できなくなる。

【００１７】単純計算で図形数の確認をしてみる。１６Ｍｂのメモリが、１Ｍｂを単位として、単純な繰り返しになっているとし、クリップ処理は、１Ｍｂの単位毎に、１６ケ所で行なうこととする。最外周を残す目的であるから、１Ｍｂ分のパターンの繰り返しは、クリップの領域矩形と交差しており、繰り返しを展開した全てのビットのパターンに関して、クリップ領域の内側か、外側かを判定する処理となる。また、クリップ結果のデータ量は、１Ｍｂの各ビットが単純な矩形のアレイ状に並んでいるとすると、１Ｍｂは、1024^2 であることから、最外周部分のビット数は、４０９２となる。即ち、１Ｍｂ分の繰り返し指示を持つ１ビット分のデータが、最外周部分クリップ後、繰り返し指示の無い４０９２ビット分のデータとなる。このようなクリップに関する問題を解決して効率の良い処理を行うことも電子線描画装置用データ作成に取って重要な問題である。

【００１８】

【発明が解決しようとする課題】上述のごとく、電子線描画装置用データ作成装置およびその方法にはいくつかの問題があった。第１は、クライアントとサーバー間のネットワーク転送負荷を削減する問題である。従来のクライアント・サーバーモデルの並列処理システムでは、全体のパターンデータを分割し、分割したデータをその都度サーバーに送って演算処理を行わせていたが、分割データの数が膨大になってクライアントとサーバー間の通信量が増え、ネットワーク転送負荷が大きくなって処理効率が低下するという問題があった。

【００１９】第２の問題は、ソフトウェアの機能追加や改良に対処する問題である。従来、ＬＳＩパターンデータのデータ処理を行いたい場合は、必要に応じたソフトウェアを作成し、それを実行することで対応したり、市販のＣＡＤツールや検証ツールを用いていたが、前者の場合は、以前作成したソフトウェアで対応できない場合には、ソフトウェアに機能の追加を行ったり、または新たにソフトウェアを作成することで対応しなければならず、作業と保守管理に負担が大きくなる問題があり、後者の場合、データ処理専用のツールではない為、実行時間が遅かったり、対応できないデータ処理が有るなどの問題があった。

【００２０】第３の問題は、繰り返し情報を持つ図形の分割に関する問題で、従来機械的に行われている繰り返し情報を持つ図形の処理を、極力繰り返し情報を保存するようにして処理の効率化を図るようにするという問題があった。

【００２１】本発明はこれらの点を解決し、クライアントとサーバー間のネットワーク転送負荷を削減し、ソフトウェアの機能追加や改良に対する処理容易にし、かつ繰り返し情報を有する図形の分割処理を効率的にした描画データ作成システムと描画データ作成方法の実現を課題とする。

【００２２】

【課題を解決するための手段】上記目的を達成するため、本発明は、ネットワークで接続された複数のワークステーションまたはパソコンを含んで構成されたクライアント／サーバー・モデルの複数のサーバーによる並列処理機能を備え、パターンデータを処理して描画データを作成する描画データ処理システムとそのような描画データの処理方法において、予め全体のパターンデータを前記複数のサーバーに転送しておき、クライアントが前記サーバーに処理させたい部分の指定情報を転送し、前記サーバーは前記指定情報で指定された部分の図形データを転送済の前記全体のパターンデータから抽出して図形演算処理等を行い、その結果を前記クライアントに転送し、前記クライアントはその結果をマージして最終的な描画用データファイルを作成することを特徴とする。

【００２３】また、ネットワークで接続された複数のワークステーションまたはパソコンを含んで構成されたクライアント／サーバー・モデルの複数のサーバーによる並列処理機能を備え、パターンデータを処理して描画データを作成する描画データ処理システムとそのような描画データの処理方法において、ソフトウェアによるデータの読み込み、データの出力の個々の転送過程で、基本情報単位に対する選択条件と変更条件を指定することにより、前記選択条件と一致する基本情報単位を前記変更条件の内容に応じて変更することを特徴とする。

【００２４】また、パターンデータを処理して描画データを作成する描画データ処理システムとそのような描画データの処理方法において、計算機上のファイルとして表現されている、繰り返し情報を含むパターンデータ群を、注目したい領域で切り取り抽出する際、抽出領域内に極力もとの前記繰り返し情報を保存するように処理し、前記繰り返し情報を繰り返し単位で処理することを特徴とする。

【００２５】

【発明の実施の形態】以下、本発明にかかる描画データ作成システムおよび描画データ作成方法を添付図面を参照にして詳細に説明する。

【００２６】まず、本発明の第１の発明では、クライアントとサーバー間のネットワーク転送負荷を削減する問題に次のように対処する。すなわち、クライアントから最初に、データ全体をサーバーに転送してしまい、その後、各サーバーに演算処理させたい部分に関する記憶領

域の位置及び大きさの情報を指定する。各サーバーは、その指定された部分の図形データを予め転送された全体データから抽出し、演算処理を行い、結果をクライアントに転送する。クライアントは各サーバーから送られたデータを出力データにマージする処理を行い、全ての部分の処理が終了すれば最終的な出力データが完成する。

【００２７】つまり、クライアントは個々の処理にあたって、各サーバーに実際の図形データは転送せずに、非常にデータ量の少ない演算部分指定情報データを転送するだけである。したがって、並列処理実行時のクライアントとサーバー間の通信データの量か大幅に削減でき、またクライアントの処理負荷をも軽減することかできるため並列処理効率の改善を実現することできる。

【００２８】従来方法と転送するデータ量は同様であるが、一般に、同じデータ量のものを多数に分割して転送するよりも、一度に転送したほうが転送時のオーバーヘッドか少なく処理速度が速いことと、並列処理実行時の通信データ量の削減により、並列処理効率向上が可能となり、全体の処理時間を短縮するのが、本発明のねらいである。

【００２９】本発明の第１の発明の第１の実施の形態について説明する。この実施の形態は、図１に示すように領域分割されたデータを処理するシステムに適用されるものである。図１において、１１はクライアントマシン、１２－１、１２－２……はサーバーマシン、１３はネットワーク、１４は設計データ、１５は描画用データ、１６はチップ全体のデータ、１６－１、１６－２はチップ全体のデータ１６から分割された各々の部分を示している。

【００３０】クライアント１１の処理フローチャートは図２のようになっている。クライアント１１は、ステップ１０１で入力されたＬＳＩ設計データ１４の階層を展開し、ステップ１０２で描画単位領域にてメッシュ上に分割する処理を行い、分割したチップ全体のデータをステップ１０３で各サーバー１２－１、１２－２……に転送する。

【００３１】次に、ステップ１０４で図形処理すべき各分割領域に対し、その位置情報をサーバー１２に転送する。サーバー１２は受け取った位置情報に応じて、すでに転送済の全体データから該当領域内の図形を抽出し、図形演算処理を施し、その結果の図形データをクライアントに転送する。

【００３２】クライアント１１はサーバー１２からの図形データを受信し（ステップ１０５）、受け取ったデータより描画用データファイルを構築する（ステップ１０６）。ステップ１０４からステップ１０６までを繰り返して、全ての分割領域の処理が終了すれば（ステップ１０７）、最終的な描画データファイル１５が完成することになる。

【００３３】本発明の第１の発明の第２の実施の形態について説明する。この実施の形態は、図３に示すように階層処理による処理システムに適用されるものである。図３において、１１はクライアントマシン、１２－１、１２－２……はサーバーマシン、１３はネットワーク、１４は設計データ、１５は描画用データ、１６はチップ全体のデータ、１６－１、１６－２はセル単位に分離された各々の部分を示している。セルとは、ＬＳＩ設計を行う際、予めまとまった機能を備えたレイアウトパタンをさす。階層設計において標準的に使用する。

【００３４】クライアント１１の処理フローは図４のようになっている。クライアント１１は、ＬＳＩ設計データの各セルに対し、独立して処理を行えるようにステップ１１１でセル間のオーバーラップを除去する等の処理を行い、処理した後のデータを各サーバー１２に転送する（ステップ１１２）。

【００３５】次にサーバー１２に処理させるセルの情報をサーバーに転送する（ステップ１１３）。サーバー１２はその情報に応じて、ステップ１１２で転送済の全体データから該当セル内の図形を抽出し、図形演算処理を施し、その結果の図形データをクライアント１１に転送する。クライアント１１はサーバー１２からの図形データを受信し（ステップ１１４）、サーバーから受け取ったデータより描画用データファイルを構築する（ステップ１１５）。ステップ１１３からステップ１１５を繰り返し、全てのセルの処理が終了すれば（ステップ１１６）、最終的な描画データファイルが完成する。

【００３６】本発明の第２の発明は、ソフトウェアの機能追加や改良に対処する第２の問題には次のように対処する。すなわち、第２の発明ではデータ処理において、一致条件と変更条件を指定し、一致条件に該当するＬＳＩパターンデータを選択し、選択されたデータを変更条件のように変更する機能を持たせることによって、ソフトウェアの機能追加や改良に対処するようにした。このような方法によりＬＳＩパターンデータ処理の種類だけ必要となったいたソフトウェアを作成する必要をなくし、従来からあるデータに一致条件と変更条件を記述するだけで所望のＬＳＩパターンデータ処理を可能にするのが、本発明のねらいである。

【００３７】具体的には、

１　ＧＤＳＩＩ　ＳＴＲＥＡＭ等のＬＳＩパターンデータファイルをソフトウェアが読み込む、あるいは出力する際に、個々の転送過程の図形、セル、セル引用、及びプロパティに対する選択条件と、変更条件を設定しておくことで条件にあったデータの選択や変更を行なう。

２　このとき、選択条件と変更条件はＡＳＣＩＩ文字列で表現し、条件毎に文字列が決まっており、それらを決まられた規則に従って羅列することで、条件を自由に設定及び変更できるようにする。

３　また、選択条件を複数指定することにより、論理積及び論理和条件指定かできるようにする。

4　また、変更条件を複数指定することにより、１度に複数のデータ変更かできるようにする。

5　このような上記データ変更は単体で実行するほかに、ＬＳＩパターンデータを入出力とするソフトウェアの前処理や後処理に組み込んで使うことができるようにする。

【００３８】以上に述べたような、選択条件と変更条件をから成る条件をフィルター的な機能として用意し、そのフィルターを通してＬＳＩパターンデータを読み込み、あるいは出力するだけで、所望のデータの選択や変更を簡単に、且つ効率的に実行するようにした。

【００３９】ここでＬＳＩパターンデータとＬＳＩパターンデータ処理について簡単に触れておく。ＬＳＩパターンデータはＬＳＩ設計ＣＡＤから出力されるデータで、ＧＤＳＩＩ　ＳＴＲＥＡＭ　ＦＯＲＭＡＴが一般的である。図形情報、セル引用情報、補足情報などの基本情報単位（以降エレメントと呼ぶ）から成り、その単位の組合せによりセルが構成され、また、そのセルの組合せによりＬＳＩパターンが構成される。エレメント、例えば図形情報は、図形種、レイヤー番号、データタイプ番号、座標値などの基本データ単位（以降レコードと呼ぶ）から構成されている。ＬＳＩパターンデータは最終的にマスタ描画装置の入力フォーマットに従いＥＢ（Electoron Beam）データに変換されることになる。また、ＬＳＩパターンデータ処理は、例えばエレメントの１つであるバウンダリは、図形種、レイヤー番号、データタイプ番号、座標値のレコードから構成される。ＬＳＩパターンデータ処理とはそれらの値を変更することである。本発明においては、データ処理はエレメント単位、またはレコード単位で実行され、レコードを１つずつ読み、次の処理へ渡す過程で可能な処理に限定される。変更する目的として、レイヤー番号を用いた半導体製造工程別のパターンのグルーピング・レイヤー番号を用いた半導体製造工程に不必要なパターンの削除、セル引用情報を用いたセル配置の変更等がある。

【００４０】次に、本発明の一致条件と変更条件について述べる。一致条件はエレメント及びレコードが持つ各情報とのマッチング条件であり、一致したものが処理の対象となる。変更条件はエレメント及びレコードが持つ情報の変更指示である。図５および図６に一致条件を、図７および図８に変更条件を例示する。

【００４１】次に一致条件と変更条件の基本動作をいくつかの例を取って説明する。一致条件の始まりを@、変更条件の始まりを&、とした場合、

@０　＆１

は、レイヤー番号０のレコードを１に変更することを表す。一致条件は省略可能で

@＆１

とすれば、全てのレコードが一致条件にマッチングしとことになる。反対に変更条件も省略可能で

@０

とすれば、一致条件でマッチングはするだけで何も処理されない。

【００４２】このような基本動作の基本フローチャートを図９に示す。まずステップ２０１でレコード単位データを読み出すと、ステップ２０２でこのレコードが一致条件にマッチングするものかどうかを調べ、マッチングした場合はステップ２０３で変更条件を実行し、ステップ２０４で変更実行後のレコードを置き換える。一致条件にマッチングしなければそのままとする。

【００４３】この場合、基本的に一度、一致条件を通過すると、データ処理は終了である。ただし、本発明では次のように一致条件と変更条件を繰り返しを許しており、レイヤー番号０のレコードを持つエレメントからレイヤー番号１とレイヤー番号２の２個のエレメントを作成したい場合は、次の一致条件に再度エレメントを渡す指示をしないといけない。それがcontinueである。

@０ copy continue　＆１　@０　＆２

【００４４】この場合のフローチャートを図１０に示す。まずステップ２１１でレコード単位データを読み出すと、ステップ２１２でこのレコードが一致条件にマッチングするものかどうかを調べ、マッチングした場合はステップ２１３で変更条件を実行し、変更実行後のレコードを置き換える。ここで一致条件にマッチングしなければそのまま次に進む。

【００４５】次にステップ２１４で一致条件にcontinueが含まれているかどうかを判定し、ふくまれている場合は再びステップ２１２に戻ってこのレコードがcontinueの後の一致条件にマッチングするかどうかを調べる。最終的に、ステップ２１５で変更実行後のレコードを置き換える。この機能を逆に利用して、

@０　@１　&delete

とすればレイヤー番号０と１以外のエレメントは削除されることになる。

【００４６】次に、一致条件での論理積条件の例について説明する。この場合は、ただ条件を空白で区切り羅列するだけでよい。例えば次の例は、レイヤー番号０のレコードを持つＢＯＵＮＤＡＲＹをレイヤー番号１に変更することを表している。

@０ boundary ＆１

【００４７】次に、一致条件での論理和条件の例に付いて説明する。先の論理積での空白の代わりにorで区切り羅列するだけでよい。例えば次の例は、ＢＯＵＮＤＡＲＹとＰＡＴＨのレイヤー番号を１に変更することを表している。

@boundary or path　&1

【００４８】次に条件文をコマンドライン上で指定する場合の例について述べる。例えば一致条件と変更条件を解釈して実行するだけのコマンドconvert があった場合、入力ＬＳＩパターンデータファイルと条件文をコマ

ンドライン上に記述し、ＬＳＩパターンデータ処理を行う方法である。このフローチャートを図１１に示す。図１１での条件実行とは図９や図１０で説明した一致条件と変更条件の実行フローである。

【００４９】図１１で、ＬＳＩレイアウトデータ１４をステップ２２１でレコード単位に入力し、ステップ２２４でコンソールなどから入力されるコマンドライン上の条件にしたがって、ステップ２２２で条件実行を行う。実行終了後結果をレコード単位で出力し、ＬＳＩレイアウトデータ１４に入力する。図１１でのステップ２２２の条件実行とは図９や図１０で説明した一致条件と変更条件の実行フローである。

【００５０】次に条件文をファイルで指定する場合の例について述べる。先の例で挙げた条件文をファイルに記述し実行時に指定するか、デフォルトで例えばcond. txtを読み込むようにする。再利用と共用が容易になる。このフローチャートを図１２に示す。

【００５１】次に条件文をあるプログラムの入力フィルタとして使用する場合の例について述べる。例えば図１３のように図形演算プログラムの前に必要なレイヤー番号のパターンのみを入力し、そのパターンについてのみ図形演算を行うことができる。このようにあるプログラムの入力フィルタとして利用が可能である。

【００５２】本発明の第３の発明は、繰り返し情報を持つ図形の分割にたいして次のように対処する。まず、計算機上のファイルとして表現されている、繰り返し情報を含む図形パターンデータ群を、注目したい領域で切り取る際に、極力もとの繰り返し情報を保存するようにすることで、以降の処理への負担を避けるように工夫する。

【００５３】ここで、本発明の第３の発明の具体的内容＊についてのべる。先ず第１はクリップ処理によるデータ増加を回避する方法である。繰り返し情報を持つ図形のクリップに際しては、展開した全ての図形に関して、内側、外側、クリップ対象の判定をせず、繰り返し情報である、単位繰り返し図形の占める領域座標、Ｘ方向及びＹ方向の繰り返しピッチ、Ｘ方向及びＹ方向の繰り返し数と、クリップ領域矩形の関係をみて、繰り返し領域矩形の内部に包含される繰り返し情報と、繰り返し領域矩形の外部にある（接するものを含む）繰り返し情報と、繰り返し領域矩形と交差する繰り返し情報に分け（以上の作業を繰り返しのクリップと呼ぶ）、交差する繰り返し情報のみ展開し、更に詳細な図形毎のクリップ処理を行なう。こうすることで、繰り返しの中で、完全に包含される図形まで、展開しクリップ領域に対する評価を実施しなくて済むため、クリップ処理時間が節約できる。また、外周部にのこす部分も、最大４つの繰り返し情報で表現でき、展開されないため、出力データ量も節約できる。

【００５４】次に、繰り返し情報のクリップ方法についてのべる。ここでは、先にのべた繰り返しのクリップ処理の詳細について、一般化した数式を使って述べる。繰り返し及びクリップ領域に関する情報を、以下のように表す。

クリップ領域矩形座標（C_{x1}、C_{y1}、C_{x2}、C_{y2}）

繰り返し単位（左下隅）図形領域（a_{x1}、a_{y1}、a_{x2}、a_{y2}）

繰り返し数n_x、n_y

繰り返しピッチd_x、d_y

図形を繰り返した結果の領域は、

【００５５】

【数１】

$$(A_{x_1}, A_{y_1}, A_{x_2}, A_{y_2}) = (a_{x_1}, a_{y_1}, a_{x_2} + (n_x - 1)d_x, a_{y_2} + (n_y - 1)d_y)$$

【００５６】この座標値より、クリップに関わる繰り返しのインデックス位置を計算すると

※【００５７】

※【数２】

$$i_{x_1} = \left[\frac{C_{x_1} - A_{x_2}}{d_x}\right] + 1 \quad i_{y_1} = \left[\frac{C_{y_1} - A_{y_2}}{d_y}\right] + 1$$

$$i_{x_2} = \left[\frac{C_{x_2} - A_{x_1}}{d_x}\right] - 1 \quad i_{y_2} = \left[\frac{C_{y_2} - A_{y_1}}{d_y}\right] - 1$$

$$j_{x_1} = \left[\frac{C_{x_1} - A_{x_1}}{d_x}\right] - 1 \quad j_{y_1} = \left[\frac{C_{y_1} - A_{y_1}}{d_y}\right] - 1$$

$$j_{x_2} = \left[\frac{C_{x_2} - A_{x_2}}{d_x}\right] + 1 \quad j_{y_2} = \left[\frac{C_{y_2} - A_{y_2}}{d_y}\right] + 1$$

【００５８】ここで求めた数値は、クリップ領域矩形と交差している図形の繰り返し内インデックスである。ｉは、クリップ領域矩形と交差している最外周のインデックス領域であり、ｊは、クリップ領域矩形と交差している最内周のインデックス領域である。この例を、図１４に示す。繰り返しピッチが、繰り返し単位図形の幅以上である通常の繰り返しの場合、繰り返し矩形と交差するインデックスは、最大でも１つなので、ｉとｊの２つの

数値を考慮する必要はないが、図に示した例のように、繰り返しピッチが、繰り返し図形幅未満の場合は、複数のインデックスと交差する可能性があるため、２つの数値でこれを取り扱っている。

【００５９】この計算の結果、i_{x2}あるいはi_{y2}が０未満の場合、あるいはi_{x1}が、n_x 以下の場合、あるいはi_{y1}がn_y 以下の場合は、繰り返し図形の全ては、クリップ領域矩形の外側にあり、クリップ処理の必要はない。また、i_{x1}とi_{y1}が、０未満で且つ、i_{x2}が、n_x 以上且つ、i_{y2}が、n_y 以上の場合は、繰り返し図形の全ては、クリップ領域矩形の内側にあり、クリップ処理の必要はない。さらに、i_{x1}と$i_{x2}+1$が等しく且つ、i_{y1}と$i_{y2}+1$とが等しい場合、クリップ領域矩形は、繰り返しの隙間の中にあるので、これも、クリップ処理の必要はない。

【００６０】これ以外の場合、繰り返し図形は、クリップ領域矩形と交差している可能性が高く（クリップ領域矩形の辺が繰り返しの隙間を通っている場合は交差しない）、クリップ処理が必要である。

【００６１】クリップにあたっては、インデックスで、
（i_{x1}、i_{y1}、i_{x2}、i_{y2}）
より外側にあるインデックスに該当する部分を、新しい繰り返し表現に直す（トーナツ状なので最大４つ）。または、内側を残すクリップの場合は、
（j_{x1}、j_{y1}、j_{x2}、j_{y2}）
より内側のインデックスに該当する部分を、新しい繰り返し表現に直す。この２つのインデックス及び、この２つのインデックスの間に含まれるドーナツ状領域に該当するインデックスは、クリップ領域矩形に交差しているため、図形そのもののクリップ処理の対象となる。

【００６２】ここで具体的な例についてのべる。まずクリップ対象が、ＥＢ（Electoron Beam）描画データ（ＭＥＢＥＳフォーマット等）のように、図形が直接繰り返し情報を持つ場合。この場合は、直接上記繰り返しのクリップ方法を適用し、クリップ領域矩形に交差しない部分を、繰り返し情報付きの図形として出力する。クリップ領域矩形に交差する図形は、その全てについて、クリップ領域矩形で切取り出力する。

【００６３】次にクリップ対象が、ＬＳＩレイアウトデータ（ＳＴＲＥＡＭフォーマット等）のように、図形のセット（セル引用を含む）である、セルの引用に繰り返し情報を持つ場合。

（Ａ）上記繰り返しのクリップ方法を適用し、クリップ領域矩形に交差しない部分を、繰り返し情報付きセル引用として出力する。クリップ領域矩形に交差するセル引用については、その全てについて、以下の処理を行なう。引用されているセルの定義内容である、図形セット全てについて、セル引用情報による、移動ローテーション等を施した結果に対し、クリップ矩形領域に対する、内／外／交差判定を行ない、削除するもの、そのまま出力するもの、クリップするものに分類する。クリップ対象になった場合、そのデータが図形である場合は、図形を、クリップ領域矩形で切り取る作業に渡す。セル引用である場合は、それも、繰り返し情報を持つ場合があるため、そのセル引用に対し（Ａ）以降の作業をリカーシブに適用する。

【００６４】

【発明の効果】以上説明したように本発明の請求項１または請求項４の発明は、クライアント／サーバーシステムで予め、全体のデータをサーバー側に転送しておくことにより、クライアントからサーバーへ実図形データを転送する必要がなくなり、並列処理時のネットワーク転送負荷が軽減でき、処理高速化が可能である。また、クライアントの処理負荷も低減し、サーバーへの処理指令遅延の軽減により、サーバーでの処理待ち時間が削減され並列処理効率の向上が可能である。

【００６５】本発明の請求項２または請求項５の発明は、選択条件と一致する基本情報単位を変更条件にしたがって変更するようにしたので、ＬＳＩパターンデータ処理の種類だけ必要となったいたソフトウェアを作成する必要がなくなり、一致条件と変更条件を記述するだけで所望のＬＳＩパターンデータ処理が可能となった。この結果、ソフトウェアの作成工数と管理工数が従来の（１／ソフトウェア数、約１／２０）に削減出来た。また、ＣＡＤツールや検証ツールを利用してＬＳＩパターンデータ処理をすることが無くなり、その処理時間も１／１０に短縮できる。

【００６６】本発明の請求項３または請求項６の発明は、繰り返し情報を含むパターンデータのクリップ処理に際し、極力もとの繰り返し情報を保存するようにしたので、必要最低限度の繰り返し展開のみで、処理を行なうことができ、クリップ処理時間増大、出力データサイズの増大を避けることができ、さらに、出力データサイズの増大を避けることで、クリップ処理の後工程の処理時間をも節約することができる。

【図面の簡単な説明】

【図１】本発明の一実施の形態のクライアント・サーバーシステムのブロック図。

【図２】図１に示す実施の形態の処理フローチャート。

【図３】本発明の他の実施の形態のクライアント・サーバーシステムのブロック図。

【図４】図３に示す実施の形態の処理フローチャート。

【図５】本発明のソフトウェア変更処理における一致条件を示す図表。

【図６】本発明のソフトウェア変更処理における一致条件を示す図表。

【図７】本発明のソフトウェア変更処理における変更条件を示す図表。

【図８】本発明のソフトウェア変更処理における変更条件を示す図表。

15

【図９】本発明のソフトウェア変更処理の一実施例のフローチャート。

【図１０】本発明のソフトウェア変更処理の他の実施例のフローチャート。

【図１１】本発明によるパターンデータ処理の一実施例のフローチャート。

【図１２】本発明によるパターンデータ処理の他の実施例のフローチャート。

【図１３】本発明によるパターンデータ処理の他の実施例のフローチャート。

【図１４】本発明による繰り返し領域及びクリップ領域

16

に関するインデックスを示す説明図。

【図１５】従来のクライアント・サーバーシステムのブロック図。

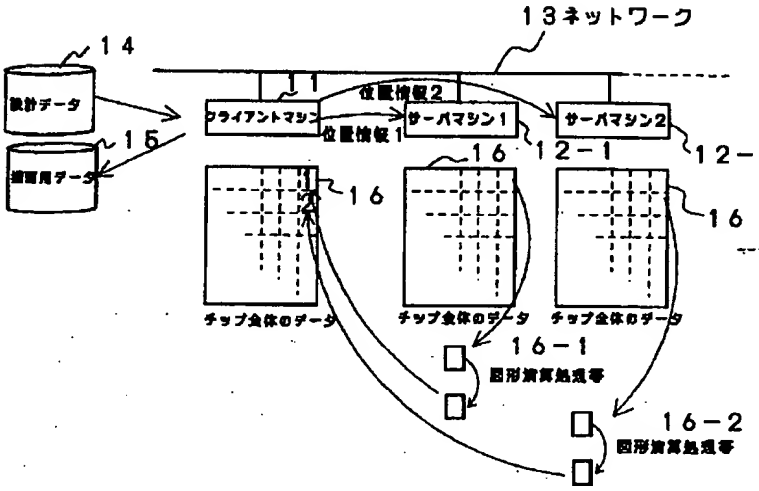
【図１６】図１５に示す例の処理フローチャート。

【図１７】繰り返し情報を持つパターンのクリップを示す説明図。

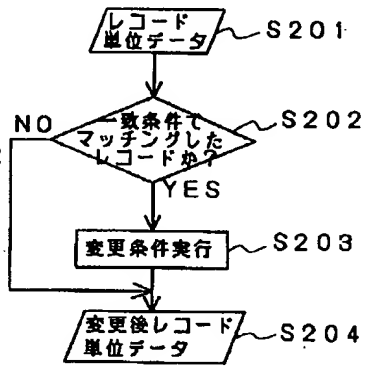
【符号の説明】

１１…クライアントマシン、１２…サーバーマシン、１３…ネットワーク、１４…設計データ、１５…描画用データ、１６…チップ全体のデータ、１６－１、１６－２…分割された各部データ。

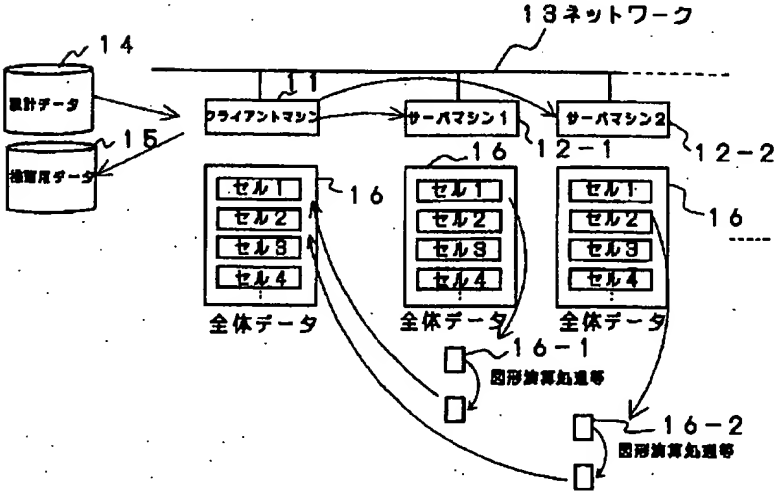
【図１】



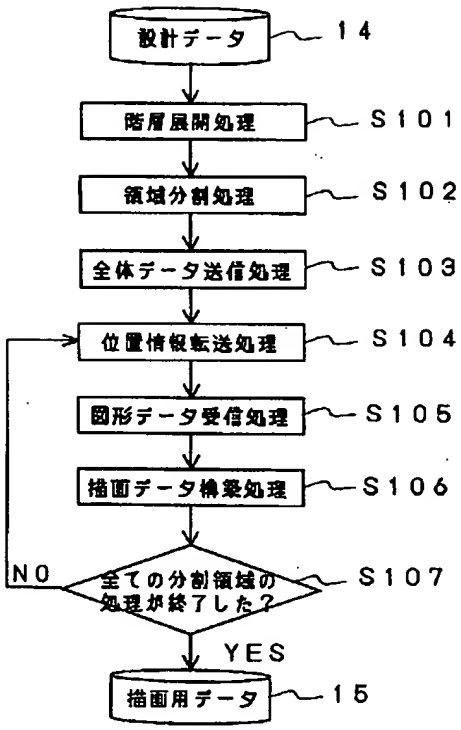
【図９】



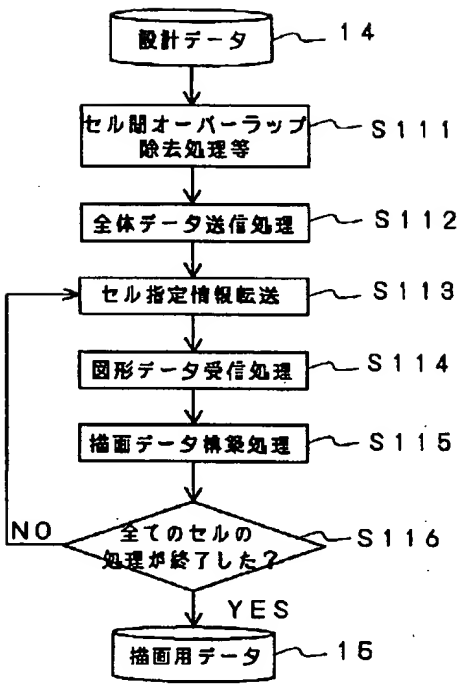
【図３】



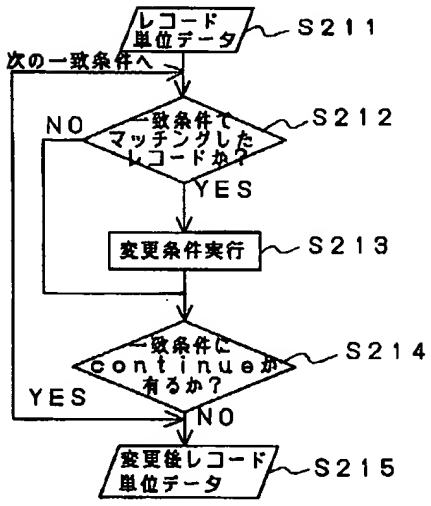
【図２】


設計データ
14
階層展開処理
S101
領域分割処理
S102
全体データ送信処理
S103
位置情報転送処理
S104
図形データ受信処理
S105
描画データ構築処理
S106
NO
全ての分割領域の処理が終了した？
S107
YES
描画用データ
15

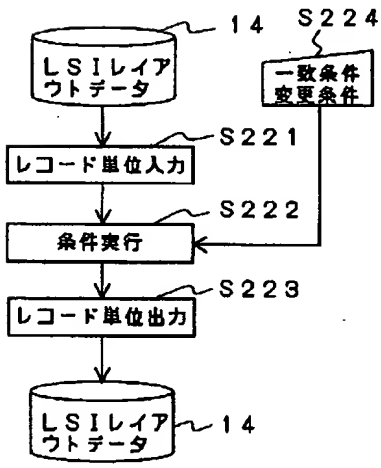
【図４】


設計データ
14
セル間オーバーラップ除去処理等
S111
全体データ送信処理
S112
セル指定情報転送
S113
図形データ受信処理
S114
描画データ構築処理
S115
NO
全てのセルの処理が終了した？
S116
YES
描画用データ
15

【図１０】


レコード単位データ
S211
次の一致条件へ
NO
一致条件でマッチングしたレコードか？
S212
YES
変更条件実行
S213
一致条件にcontinueが有るか？
S214
YES
NO
変更後レコード単位データ
S215

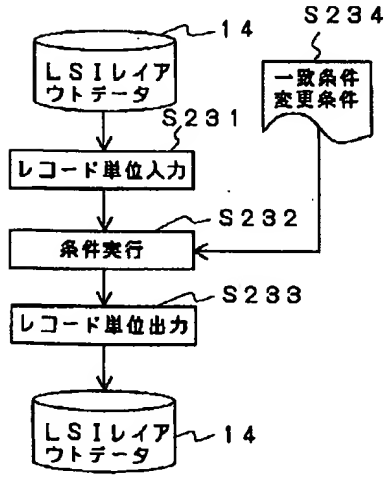
【図１１】


14
LSIレイアウトデータ
S224
一致条件変更条件
S221
レコード単位入力
S222
条件実行
S223
レコード単位出力
LSIレイアウトデータ
14

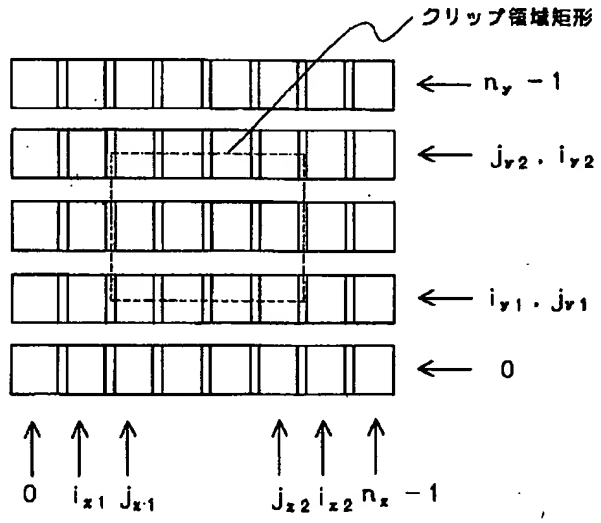
【図５】

一致条件	該当レコード又はエレメント
layer[-layer][;datatype[-datatype]]	指定したレイヤー番号及びデータタイプ番号を持つエレメント．- を用いることで範囲指定も可能とする．ここで [] で囲まれた文字列は省略可能とする．
nonref	セル引用情報以外のエレメント．
boundary, bnd	バウンダリエレメント (以降 BOUDARY と呼ぶ)．
path,	パスエレメント (以降 PATH と呼ぶ)．
sref	セル引用情報の Single REFlection エレメント (以降 SREF と呼ぶ)．
aref	セル引用情報の Array REFlection エレメント (以降 AREF と呼ぶ)．
text	テキストエレメント (以降 TEXT と呼ぶ)．
string s	文字列 s に一致する ASCII 文字列情報 (以降 STRING と呼ぶ) レコード，又はそのレコードを持つ TEXT
wstring s	ワイルドカード指定の文字列 s に一致する STRING レコード，又は TEXT．ワイルドカードは任意の文字列は ？，任意の文字列の 0 回以上の繰り返しは * で表現する．
node	ノードエレメント (以降 NODE と呼ぶ)．
box	ボックスエレメント (以降 BOX と呼ぶ)．
ref	セル引用情報エレメント (以降 SREF/AREF と呼ぶ)．

【図１２】



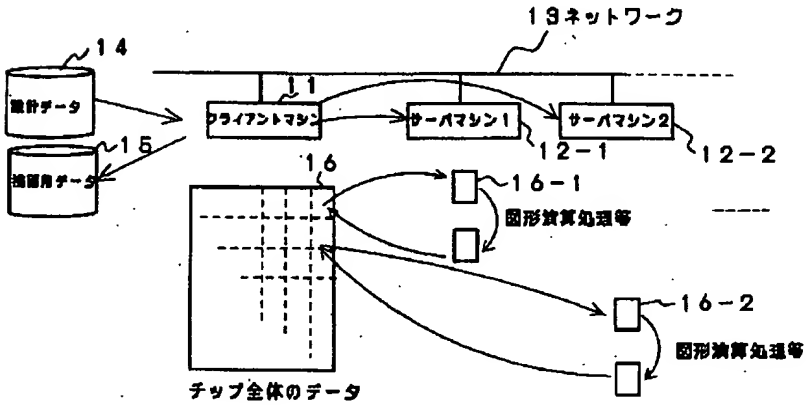
【図１４】



【図６】

一致条件	該当レコード又はエレメント
sname s	文字列　s　に一致する引用セル名レコード(以降 SNAME と呼ぶ), 又は SREF/AREF.
width f($f \geq 0$)	実数 f と一致する PATH 幅レコード, 又はそのレコードを持つ PATH/TEXT.
pbext f, peext f	実数 f と一致するパスの始端と終端のオーバーハングレコード, 又はそのレコードを持つ PATH/TEXT.
ptoh, ptnoh, ptrnd, ptve	パスの始端と終端のオーバーハングタイプレコード, 又はそのレコードを持つ PATH/TEXT の選択.
reflect b	ブーリアン b と一致するミラーリングレコード, 又はそのレコードを持つ SREF/AREF/TEXT.
absang b, absmag b	ブーリアン b と一致する絶対回転角度表現レコード, 絶対倍率表現レコード, またはそのレコードを持つ SREF/AREF/TEXT.
angle f($0 \leq f < 360$)($d \leq 1.0 \times 10^{-6}$)	実数 f と一致する回転角度レコード, 又はそのレコードを持つ SREF/AREF/TEXT.
mag f($f > 0$)($d \leq 1.0 \times 10^{-6}$)	実数 f と一致する倍率レコード, 又はそのレコードを持つ SREF/AREF/TEXT.
strname s	セル名 s に属するエレメント.
continue	次の一致条件にもレコードを渡す.

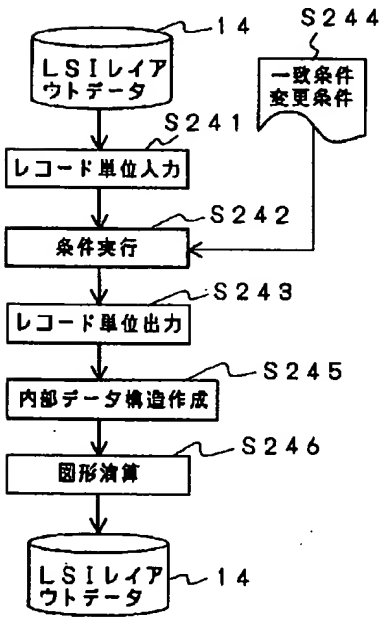
【図１５】



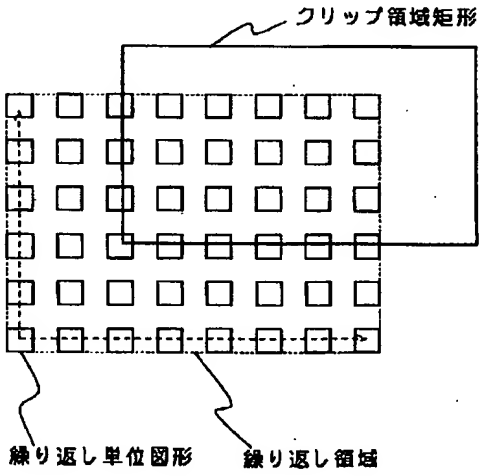
【図７】

＆変更条件	該当レコード又はエレメント
layer[;datatype]	指定したレイヤー番号及びデータタイプ番号に変更.
boundary, bnd	PATH の場合はアウトライン処理を行い BOUNDARY に変換する.BOX の場合はそのまま矩形の BOUNDARY に変換する.
path	NODE を PATH に変換する.
text	TEXT の STRING が "SNAME Nx, Ny, Dx, Dy" のようなフォーマットになっていた場合, SREF/AREF に変換する. Nx:X 方向繰り返し数, Ny:Y 方向繰り返し数, Dx:X 方向繰り返し間隔, Dy:Y 方向繰り返し間隔
sname s	SNAME を s に変更する.
ref	SREF/AREF に変換する.
string s	STRING を s に変換する.
width f($f \geq 0$) pbext f, peext f ptoh, ptnoh, ptrnd, ptve	PATH の WIDTH を f に変更する. パスの始端と終端のオーバーハングを f に変更する. パスの始端と終端のオーバーハングタイプを変更する.
divide-zapath	パスの辺と辺の間の角度がゼロの場合, 折り返し点を境にし, 2 つの PATH に分割する.

【図１３】



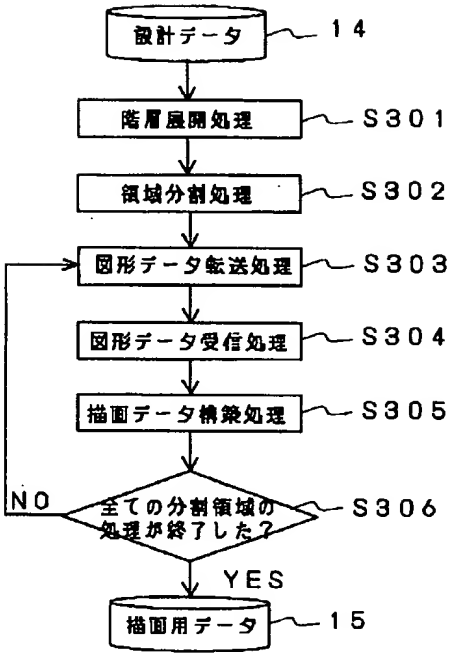
【図１７】



【図８】

＆変更条件	該当レコード又はエレメント
extend-zwpath f($f \neq 0$)(width=0)	WIDTH=0 のパスの始点と終点の座標値を f の分だけ延長する.
resize f($f \neq 0$)(width>0)	リサイズ処理.
reflect b, absang b, absmag b	b に変更.
angle f($0 \leq f < 360$), mag f($f > 0$)	f に変更
afold n, acorner n, acornerh n, aorigin n, aorigin h	AREF を指定された方法で n 個数だけ展開する.
round f($f > 0$) rotate b f f (reflect,angle,mag) move f f (dx,dy)	座標値を f で丸め処理 座標値を指定されたローテーション情報に従い変換. 座標値を dx, dx の分だけ移動.
copy	エレメントをコピーする.
delete	エレメントを削除する.
pdelete	プロパティを削除する.
dump	エレメント情報を標準出力する.
message s	メッセージ s を標準出力する.
abort s	処理を強制終了させる.
continue	エレメントをコピーし, コピーの方は次の一致条件へ渡され, オリジナルの方はそのまま変更条件を続ける.

【図１６】



フロントページの続き

(51) Int. Cl.[6]	識別記号	ＦＩ	
		Ｈ０１Ｌ　21/30	５０２Ｇ
			５４１Ｃ